(19) KOREAN INTELLECTUAL PROPERTY OFFICE

KOREAN PATENT ABSTRACTS

(11)Publication number: 1020030046581 A
(43)Date of publication of application: 18.06.2003

(21)Application number: 1020010076442
(22)Date of filing: 05.12.2001

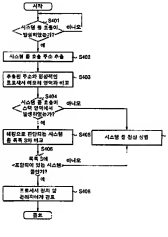
(71)Applicant: ELECTRONICS AND TELECOMMUNICATIONS RESEARCH INSTITUTE

(72)Inventor: CHOI, YANG SEO
PARK, CHI HANG
SEO, DONG IL
SON, SEUNG WON

(51)Int. Cl H04L 12 /22

(54) REAL TIME BUFFER OVERFLOW HACKING DETECTING METHOD

(57) Abstract:



PURPOSE: A real time buffer overflow hacking detecting method is provided to detect and prevent a buffer overflow hacking attempt to a system by analyzing a system call generation position on a real time basis and detecting an unknown hacking form. CONSTITUTION: It is judged whether a system call paging has occurred(S401). If the system call paging has occurred, a system call generation address is extracted(S402). The extracted address is compared to a normal process memory region(S403). It is judged whether the system call paging has occurred in a stack region of a memory(S404). If the system call paging has occurred in the stack region of the memory, the system call paging is compared with a system call list(S405), to judge whether it is on the system call list(S406). If the system call paging has not occurred in the stack region of the memory, the system call is normally processed(S407). If the system call is on the system call list, a corresponding process is stopped and an alarm is provided to a system manager(S408).

Legal Status
Date of request for an examination (20011205)
Notification date of refusal decision (00000000)
Final disposal of an application (registration)
Date of final disposal of an application (20060228)
Patent registration number (1005601660000)
Date of registration (20060306)
Number of opposition against the grant of a patent ( )
Date of opposition against the grant of a patent (00000000)
Number of trial against decision to refuse (2004101006280)
Date of requesting trial against decision to refuse (20041230)

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.
H04L 12/22

(11) 공개번호 특2003-0046581
(43) 공개일자 2003년06월18일

(21) 출원번호 10-2001-0076442
(22) 출원일자 2001년12월05일

(71) 출원인 한국전자통신연구원
대한민국
305-350
대전 유성구 가정동 161번지

(72) 발명자 최양서
대한민국
305-390
대전광역시유성구전민동306-8번지301호

서동일
대한민국
305-390
대전광역시유성구전민동464-1번지엑스포아파트107동501호

손승원
대한민국
305-390
대전광역시유성구전민동엑스포아파트208동902호

박치항
대한민국
305-333
대전광역시유성구어은동한빛아파트131동1002호

(74) 대리인 이화익
권태복

(77) 심사청구 있음

(54) 출원명 실시간 버퍼 오버플로우 해킹 탐지 방법

요약

본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법은, 미리 결정된 프로그램 목록에 속하는 프로그램으로부터 시스템 콜이 발생되면, 발생된 시스템의 콜의 발생 위치를 추출하고, 추출된 시스템 콜 발생 위치와 시스템 콜 자체를 이용한 기 결정된 해킹으로 판단될 수 있는 시스템 콜과 비교 분석하여 발생된 시스템 콜이 해킹으로 판단될 수 있는 시스템 콜에 포함되는 경우 해킹으로 판단하여 해당 프로그램을 정지시키고 관리자에게 경보를 보내는 것이다. 따라서, 버퍼 오버플로우 취약점을 가지고 있는 프로그램을 사용하더라도 해킹 시도를 실시간으로 탐지하여 방어할 수 있으며, 알려지지 않은 해킹 방법 역시 스택 영역에서 수행되는 경우 탐지할 수 있어 버퍼 오버플로우 해킹 시도를 효과적으로 방지할 수 있는 것이다.또한, 시스템 해킹시 가장 널리 사용되고 있는 버퍼 오버플로우 해킹을 방지함으로써, 보다 안전하고 높은 수준의 시스템 보안 강도를 유지할 수 있는 것이다.

대표도

도5

색인어

오버플로우, 버퍼, 실시간, 해킹, 탐지, 시스템콜,렁파일러, 클래스

명세서

도면의 간단한 설명

도 1은 일반적인 하드웨어 시스템에 대한 블록 구성을 나타낸 도면.

도 2는 버퍼 오버 플로우 해킹 문제를 설명하기 위한 일반적인 프로그램 수행시 프로그램에 할당되는 메모리 구조를 나타낸 도면.

도 3은 일반적인 버퍼 오버플로우 공격 기법을 설명하기 위한 메모리 구조를 나타낸 도면.

도 4는 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법을 구현하기 시스템 상태를 나타낸 도면.

도 5는 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법에 동작 플로우차트를 나타낸 도면.

도면의 주요 부분에 대한 부호의 설명

350 : 공격 결정 클래스000 360 : 시스템 콜 탐지 엔진

370 : 시스템 콜 주소 추출 모듈

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 실시간 버퍼 오버플로우 해킹 탐지 방법에 관한 것으로서, 특히 시스템 내부에서 발생되는 시스템 콜의 발생 위치를 실시간으로 분석하고, 알려지지 않은 해킹 형태에 대한 탐지가 가능하도록 함으로써, 시스템에 대한 버퍼 오버 플로우 해킹 시도를 탐지하고 방어할 수 있도록 한 실시간 버퍼 오버플로우 해킹 탐지 방법에 관한 것이다.

일반적으로 전산망을 이용하는 모든 이용자들에게는 전산망에 자신이 접근할 수 있는 권한 영역이 정해져 있다. 그러나, 이러한 이용자들이 이용할 수 있도록 허락한 소프트웨어 내부에 버퍼가 오버플로우될 수 있는 결함이 존재하게 되면, 고의로 조작된 값을 결함이 존재하는 소프트웨어에 입력시킴으로서, 권한 밖의 명령을 수행하거나 보다 상위의 이용 권한을 획득하여 허가되지 않은 작업을 수행하여 불법으로 정보를 취득하고 시스템을 파괴 또는 변조할 수 있다.

도 1은 일반적인 하드웨어 시스템에 대한 블록 구성을 나타낸 도면으로서, 중앙 처리 장치(11), 중앙 처리 장치(11)에 연결된 주 기억 장치(12), 주 기억 장치(12)에 연결된 보조 기억 장치(13), 중앙 처리 장치(11)에 연결된 입력 장치(14) 및 출력 장치(15)를 구비한다.

여기서, 하드웨어 시스템은, 컴퓨터의 전체 동작을 제어하고, 관리하는 중앙 처리 장치(11), 상기 중앙 처리 장치(11)에서 수행되는 프로그램을 저장하고 작업 수행 중 이용되는 또는 작업 수행중에 발생되는 각종 데이터를 저장하는 주 기억장치(12)와 보조 기억장치(13) 및 사용자와 데이터 입출력을 위한 입출력장치(14, 15)를 포함한다.

그리고, 보조 기억 장치(13)는 대량의 데이터를 저장하는 역할을 하며, 상기 입출력 장치(14, 15)는 일반적인 키보드, 디스플레이장치 및 프린터 등을 포함한다.

상기와 같은 하드웨어 시스템의 주 기억 장치(12)에는 소프트웨어 내부에 존재하는 버퍼 오버 플로우가 일어날 수 있는 결함을 이용하는 해킹을 막기 위해 버퍼 오버플로우의 결함을 미리 검출하기 위한 프로그램이 저장되어 있으며, 중앙 처리 장치(11)의 제어에 따라 수행된다.

상기한 버퍼 오버 플로우의 결함을 검출하는 기존의 방법에 대해서는 국내특허출원 1999-058299호 상세하게 제시되어 있다.

상기 제시된 종래 기술에 대하여 간단하게 살펴보기로 하자.

먼저, 점검할 소스 파일 이름을 입력 받아 점검할 변수를 리스트로 작성하여 관리하고, 상기 리스트에 있는 변수를 사용하였는지를 검사한다.

그리고, 사용자로 부터 입력된 문자열 변수의 크기를 점검하는지를 추적하여 프로그램 내의 버퍼 오버플로우에 대한 결함 존재 유무를 검출하게 되는 것이다.

이러한 종래의 방법은, 버퍼 오버플로우를 이용한 해킹을 막기 위해 프로그램 제작 후 소스 코드에 대해 고의적 버퍼 오버플로우가 일어날 수 있는 부분을 미리 검출함으로써, 소프트웨어 테스트 단계의 여러 측면에서 치명적인 버퍼 오버플로우 결함을 미리 찾아내어 이로 인한 패치 비용을 절감하고, 해당 소프트웨어의 이미지 실추를 방지할 수 있는 것이다.

그러나, 이러한 방법은 프로그램 소스 코드를 통해 버퍼 오버플로우 취약점을 가지고 있는 함수의 사용 여부를 확인하여 판단하는 방식으로 버퍼 오버플로우 방지를 수행하기 위해서는 프로그램 소스 코드가 필요하고, 프로그램 수행중에 발생하는 실시간 버퍼 오버플로우 해킹을 방지하는 것이 불가능하다는 문제점을 안고 있다.

발명이 이루고자 하는 기술적 과제

따라서, 본 발명은 상기한 종래 기술에 따른 문제점을 해결하기 위하여 안출한 것으로, 본 발명의 목적은, 시스템 내부에서 발생되는 시스템 콜의 발생 위치를 실시간으로 분석하고, 알려지지 않은 해킹 형태에 대한 탐지가 가능하도록 함으로써, 시스템에 대한 버퍼 오버 플로우 해킹 시도를 탐지하고 방어할 수 있도록 한 실시간 버퍼 오버플로우 해킹 탐지 방법을 제공함에 있다.

특히, 시스템 콜의 메모리상의 위치를 오직 현재 발생한 시스템 콜만을 수집하여 분석하기 때문에 지속적으로 관리해야 하는 시스템 콜의 개수가 단위 시간에 오직 1개 뿐이고 알려지지 않은 버퍼 오버플로우 해킹에 대한 방지가 가능하도록 한 방법을 제공하는 것이다.

발명의 구성 및 작용

상기한 목적을 달성하기 위한 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법의 일측면에 따르면, 시스템 콜 호출이 발생한 경우, 발생된 시스템 콜을 입력으로 하여 발생된 시스템 콜의 발생 위치(주소)를 추출하는 단계; 시스템 콜 호출 위치가 추출되면, 추출된 주소(위치)와 정상적인 프로세스 메모리 영역을 비교하여 시스템 콜 호출이 메모리의 스택 영역에서 발생하였는지를 판단하는 단계; 판단 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생한 경우, 기 설정된 해킹으로 판단할 수 있는 시스템 콜 목록과 비교하여, 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되어 있는 시스템 콜인지를 판단하는 단계; 상기 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되는 경우 해당 프로세스를 정지시키고 시스템 관리자에게 경보를 제공하는 단계를 포함할 수 있다. 여기서, 상기 정상적인 프로세스 메모리영역은, 상기 취약점을 자기고 있을 것으로 판단되는 프로그램 목록에 속하는 각각의 프로그램이 사용하는 메모리 영역이 될 수 있다.

또한, 취약점을 가지고 있을 것으로 판단되는 시스템 프로그램 목록과, 해킹으로 판단할 수 있는 시스템 콜 목록을 각각 결정하여 저장하는 단계를 더 포함할 수 있다.

또한, 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법의 다른 측면에 따르면, 취약하다고 판단되는 프로그램 및 서비스 목록을 작성하는 단계; 결정된 프로그램 및 서비스가 사용하는 메모리 주소에 대한 정보를 작성하는 단계; 해킹으로 판단할 수 있는 시스템 콜 목록을 작성하는 단계; 프로그래밍 의해 발생된 시스템 콜로 부터 시스템 콜의 메모리 상에서 위치를 추출하는 기능을 작성하는 단계; 상기 단계들에서 작성한 내용과 시스템 콜 모니터링 엔진을 병합시키는 단계; 병합된 결과를 이용하여 발생된 시스템 콜에 대한 해킹 여부를 탐지하는 단계를 더 포함한다.

한편, 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법을 수행하기 위하여 디지털 처리장치에 의해 실행될 수 있는 명령어들의 프로그램이 유형적으로 구현되어 있으며, 디지털 처리장치에 의해 판독될 수 있는 기록 매체의 일 측면에 따르면, 시스템 콜 호출이 발생한 경우, 발생된 시스템 콜을 입력으로 하여 발생된 시스템 콜의 발생 위치(주소)를 추출하는 단계; 시스템 콜 호출 위치가 추출되면, 추출된 주소와 기 설정된 취약점을 자기고 있을 것으로 판단되는 프로그램 목록에 속하는 각각의 프로그램이 사용하는 메모리 영역을 비교하여 시스템 콜 호출이 메모리의 스택 영역에서 발생하였는지를 판단하는 단계; 판단 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생한 경우, 기 설정된 해킹으로 판단할 수 있는 시스템 콜 목록과 비교하여, 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되어 있는 시스템 콜인지를 판단하는 단계; 상기 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되는 경우 해당 프로세스를 정지시키고 시스템 관리자에게 경보를 제공하는 단계를 수행한다.

이하, 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법에 대한 바람직한 실시예에 대하여 첨부한 도면을 참조하여 상세하게 살펴보기로 하자.

도 2는 버퍼 오버플로우 해킹 문제를 설명하기 위한 일반적인 프로그램 수행시 프로그램에 할당되는 메모리 구조를 나타낸 도면이고, 도 3은 일반적인 버퍼 오버플로우 공격 기법을 설명하기 위한 메모리 구조를 나타낸 도면이다.

도 2에 도시된 바와같이 프로그램이 메모리에 적재되어 수행되어질 경우, 프로그램의 실행 코드들은, 프로그램 코드 영역 즉, TEXT 영역에 적재되고, 전역 변수 및 상수들은 DATA영역에 적재된다.

또한, 정적 변수들은 BBS 영역에 적재되고, 동적으로 할당되는 변수들은 힙 영역에 할당된다.

스택 영역은 프로그램 수행중에 호출되는 함수들에서 사용되는 지역 변수와 함수 호출 및 복귀에 사용되는 정보들이 저장된다.

프로그램의 실제 수행시 스택 영역 확인을 위한 예제 프로그램은 아래의 알고리즘과 같다.

```
void function(char *str)

{

  char buffer[8];

  strcpy(buffer, str);

}

int main(void)

{

  char str[512];

  gets(str);

  function(str);

}
```

상기한 C 언어 코드의 경우 도 3과 같은 스택 메모리 구조를 갖게 된다.

즉, 도 3에 도시된 바와 같이, main함수의 지역 변수인 char str[512];는 main함수의 스택영역에 저장되어 있다.

그리고, 함수 function이 호출될 때, 함수 function의 인자로서 function 함수의 스택 영역에 str[512] 배열의 주소가 전달된다. 함수 function의 스택 영역에는 char Buffer[8];의 지역 변수가 존재하고 있다. 이때, str의 크기는 512바이트인 반면 buffer의 크기는 8바이트이다.

따라서, 512바이트의 내용이 8바이트의 크기에 덮어 써지게 되는 경우, function함수의 스택 영역에 포함되어 있는 복귀 주소 역시 덮어 써지게 된다.

function함수의 실행이 완료되면, 최초 function 함수가 호출된 곳으로 프로그램의 흐름이 되돌아 가야하는데, 저장되어 있던 복귀 주소가 변경됨으로 해서 정상적인 메모리 영역으로 복귀할 수 없게 된다. 이때, 덮어 쓰게되는 str[512]의 내용 중 복귀 주소를 덮어 쓸 부분을 잘 조정하여 원하는 곳으로 프로그램 흐름이 이동하도록 지정하면, 원하는 명령을 수행할 수 있게 된다.

흔히 공격자들은 수행하고자 하는 명령을 포함하는 쉘 코드(shell code: 공격 코드)를 작성하여 사용자의 입력이 원하는 명령을 수행하도록 한다. 이것이 가장 전형적인 버퍼 오버플로우 공격 기법이다. 이때 모든 공격자가 삽입하는 명령은 스택 영역에 위치한 변수에 저장된다는 것을 알 수 있다. 따라서, 공격자가 삽입하여 수행하는 명령은 스택 영역상에서 수행되기 때문에 실행되는 명령이 스택 영역에서 수행되는지를 확인할 수 있다면, 버퍼 오버플로우 해킹임을 판단할 수 있게 된다.

이러한 해킹을 방지하고, 방어하기 위한 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법에 대하여 살펴보기로 하자.

도 4는 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 시스템 및 그 방법을 구현하기 시스템 상태를 나타낸 도면이다.

도 4에 도시된 바와 같이, 시스템 콜 탐지 엔진(360)은 네트워크를 통해 시스템 외부에서 내부로 침입할 수 있는 개방된 서비스를 제공하는 데몬(Demon) 프로그램과 기 발표된 버퍼 오버플로우 취약점을 포함하고 있는 취약 판단 프로그램들로 구성된 목록 즉, 취약점을 가지고 있을 것으로 판단되는 프로그램 목록 P(310)와 목록 P에 포함되는 각각의 프로그램들에 대해 실제 실행시 할당되는 메모리 영역을 확인하여 얻어낸 메모리 정보 M 즉, 프로그램 목록 P에 속하는 각각의 프로그램이 사용하는 메모리 영역에 대한 정보(320), 그리고 각종 해킹 코드 및 해킹 문서 등을 통해 얻을 수 있는 해킹에 성공하기 위해 수행되어야만 하는 각종 명령들이 수행되어질 때 발생하는 해킹으로 판단되는 시스템 콜 목록 즉, 해킹으로 판단되는 취약한 시스템 콜 목록 S(330)를 병합하여 공격 결정 클래스(350)를 작성하게 된다.

공격 결정 클래스(350)는 일반 C++ 프로그램에서 사용하는 클래스로 C++ 컴파일러를 통해 컴파일 시킬 수 있는 소스 형태이다.

공격 결정 클래스(350)와 시스템 콜 탐지 엔진 구조(340), 그리고 발생되는 시스템 콜의 호출 위치를 확인할 수 있는 시스템 콜 발생 시스템 콜 주소 추출 모듈(370)을 병합하여 컴파일러를 통해 시스템 콜 탐지 엔진(360)을 구성한다.

이렇게 구성된 시스템 콜 엔진(360)은 온라인 시스템 상에서 버퍼 오버플로우 해킹 탐지의 핵심 모듈이 된다.

이러한 구성을 갖는 시스템을 이용한 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 및 이를 방어하기 위한 방법에 대하여 도 5를 참조하여 단계적으로 설명해 보기로 하자.

도 5는 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 및 이를 방어하기 위한 방법에 동작 플로우차트를 나타낸 도면이다.

도 5에 도시된 바와 같이, 실시간 버퍼 오버플로우 해킹 탐지 및 이를 방어하기 위한 방법은, 먼저 시스템 콜 호출이 발생하였는지를 판단한다(S401). 즉, 시스템 콜 호출이 발생할 때 까지 시스템 콜 탐지 엔진은 대기한다.

판단 결과, 시스템 콜 호출이 발생한 경우, 발생된 시스템 콜을 입력으로 하여 발생된 시스템 콜의 발생 위치(주소)를 추출한다(S402).

이어, 시스템 콜 호출 위치가 추출되면, 추출된 주소(위치)와 정상적인 프로세스 메모리 영역과 비교한다(S403).

비교 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생하였는지를 판단한다(S404).

판단 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생한 경우, 해킹으로 판단할 수 있는 시스템 콜 목록과 비교하여(S405), 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되어 있는 시스템 콜인지를 판단하게 되는 것이다(S406).

그러나, 상기 S404단계에서, 시스템 콜 호출이 메모리의 스택 영역에서 발생하지 않았을 경우에는 시스템 콜을 정상적으로 처리한다(S407).

상기 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되는 경우 해당 프로세스를 정지시키고 시스템 관리자에게 경보를 제공한다(S408).

그러나, 상기 S406단계에서 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되지 않는 경우 시스템 콜을 정상적으로 처리하게 되는 것이다.

상기 S408단계의 관리자에게 경보 후에는 다음 시스템 콜을 기다리게 되는 것이다.

결국, 본 발명은, 결정된 프로그램 목록에 속하는 프로그램으로부터 발생하는 시스템 콜의 발생을 감지하고, 발생된 시스템의 콜의 발생 위치를 추출한다.

시스템 콜의 발생 위치가 추출되면, 추출된 시스템 콜 발생 위치와 시스템 콜 자체를 이용한 기 결정된 해킹으로 판단될 수 있는 시스템 콜과 비교 분석하여 발생된 시스템 콜이 해킹으로 판단될 수 있는 시스템 콜에 포함되는 경우 해킹으로 판단하여 해당 프로그램을 정지시키고 관리자에게 경보를 보내는 것이다.

이러한 프로세스를 처리하기 위해서는 버퍼 오버 플로우 취약점을 이용한 해킹을 검출하는 모듈을 작성하여야 하는데, 이러한 모듈 작성 방법에 대하여 간단하게 살펴보자.

먼저, 취약하다고 판단되는 프로그램 및 서비스 목록을 결정하여 저장하고, 상기 결정된 프로그램 및 서비스가 사용하는 메모리 주소에 대한 정보를 추출하게 된다.

그리고 해킹으로 판단할 수 있는 시스템 콜을 결정하고, 프로그램에 의해 발생된 시스템 콜로부터 시스템 콜의 메모리 상에서 위치(주소)를 추출하게 된다.

상기 과정에서의 내용과 시스템 콜 모니터링 엔진을 병합시킨 후, 병합된 결과를 통해 본 발명에서 활용할 버퍼 오버플로우 탐지 및 방어 시스템을 구성하게 되는 것이다.

이렇게 구성된 버퍼 오버플로우 탐지 시스템을 도 5와 같이 운용하여 버퍼 오버플로우 해킹 시도 여부를 검출 및 판단하게 되는 것이다.

발명의 효과

상기한 바와 같이 본 발명에 따른 실시간 버퍼 오버플로우 해킹 탐지 방법은, 버퍼 오버플로우 취약점을 가지고 있는 프로그램을 사용하더라도 해킹 시도를 실시간으로 탐지하여 방어할 수 있으며, 알려지지 않은 해킹 방법 역시 스택 영역에서 수행되는 경우 탐지할 수 있어 버퍼 오버플로우 해킹 시도를 효과적으로 방지할 수 있는 것이다.

또한, 시스템 해킹시 가장 널리 사용되고 있는 버퍼 오버플로우 해킹을 방지함으로써, 보다 안전하고 높은 수준의 시스템 보안 강도를 유지할 수 있는 것이다.

(57) 청구의 범위

청구항 1.
시스템의 실시간 버퍼 오버플로우 해킹 탐지 방법에 있어서,

시스템 콜 호출이 발생한 경우, 발생된 시스템 콜을 입력으로 하여 발생된 시스템 콜의 발생 위치를 추출하는 단계;

시스템 콜 호출 위치가 추출되면, 추출된 위치와 정상적인 프로세스 메모리 영역을 비교하여 시스템 콜 호출이 메모리의 스택 영역에서 발생하였는지를 판단하는 단계;

판단 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생한 경우, 기 설정된 해킹으로 판단할 수 있는 시스템 콜 목록과 비교하여, 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되어 있는 시스템 콜인지를 판단하는 단계;

상기 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되는 경우 해킹으로 판단하여 해당 프로세스를 정지시키고 시스템 관리자에게 경보를 제공하는 단계를 포함하는 실시간 버퍼 오버플로우 해킹 탐지 방법.

청구항 2.
제1항에 있어서,

취약점을 가지고 있을 것으로 판단되는 시스템 프로그램 목록과, 해킹으로 판단할 수 있는 시스템 콜 목록을 각각 결정하여 저장하는 단계를 더 포함하는 실시간 버퍼 오버플로우 해킹 탐지 방법.

청구항 3.
제1항에 있어서,

상기 정상적인 프로세스 메모리영역은,

상기 취약점을 가지고 있을 것으로 판단되는 프로그램 목록에 속하는 각각의프로그램이 사용하는 메모리 영역인 실시간 버퍼 오버플로우 해킹 탐지 방법.

청구항 4.
제1항에 있어서,

취약하다고 판단되는 프로그램 및 서비스 목록을 작성하는 단계;

결정된 프로그램 및 서비스가 사용하는 메모리 주소에 대한 정보를 작성하는 단계;

해킹으로 판단할 수 있는 시스템 콜 목록을 작성하는 단계;

프로그래밍 의해 발생된 시스템 콜로 부터 시스템 콜의 메모리 상에서 위치를 추출하는 기능을 작성하는 단계;

상기 단계들에서 작성한 내용과 시스템 콜 모니터링 엔진을 병합시키는 단계;

병합된 결과를 이용하여 발생된 시스템 콜에 대한 해킹 여부를 탐지하는 단계를 더 포함하는 실시간 버퍼 오버플로우 해킹 탐지 방법.

청구항 5.
실시간 버퍼 오버플로우 해킹 탐지/방어 방법을 수행하기 위하여 디지털 처리장치에 의해 실행될 수 있는 명령어들의 프로그램이 유형적으로 구현되어 있으며, 디지털 처리장치에 의해 판독될 수 있는 기록 매체에 있어서,

시스템 콜 호출이 발생한 경우, 발생된 시스템 콜을 입력으로 하여 발생된 시스템 콜의 발생 주소를 추출하는 단계;

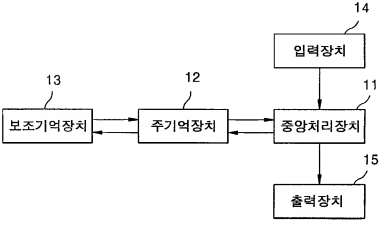
시스템 콜 호출 주소가 추출되면, 추출된 주소와 기 설정된 취약점을 가지고 있을 것으로 판단되는 프로그램 목록에 속하는 각각의 프로그램이 사용하는 메모리 영역을 비교하여 시스템 콜 호출이 메모리의 스택 영역에서 발생하였는지를 판단하는 단계;

판단 결과, 시스템 콜 호출이 메모리의 스택 영역에서 발생한 경우, 기 설정된 해킹으로 판단할 수 있는 시스템 콜 목록과 비교하여, 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되어 있는 시스템 콜인지를 판단하는 단계;

상기 판단 결과, 시스템 콜이 해킹으로 판단할 수 있는 시스템 콜 목록에 포함되는 경우 해당 프로세스를 정지시키고 시스템 관리자에게 경보를 제공하는 단계를 수행하는 기록 매체.

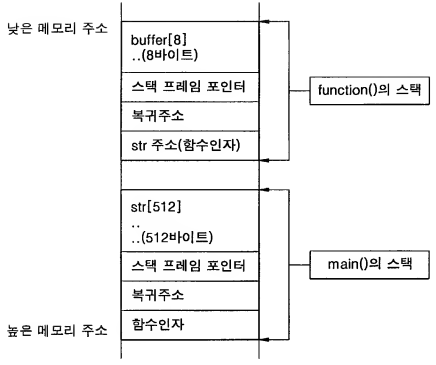
도면

도면 1

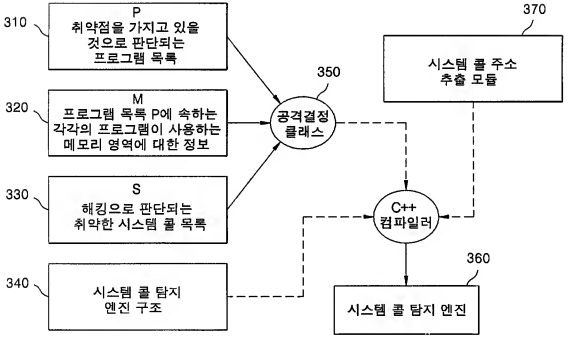


도면 2

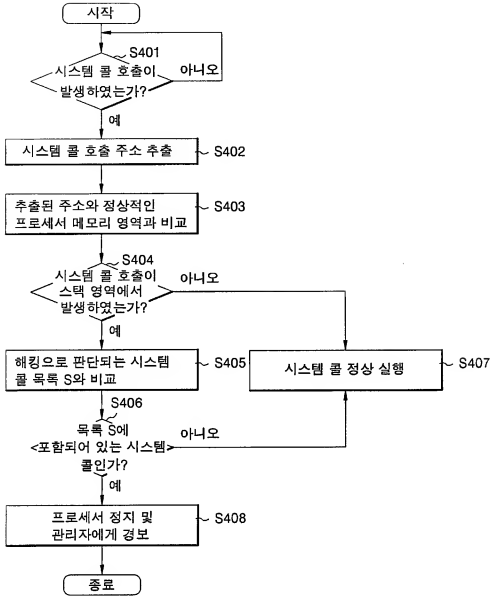
도면 3


낮은 메모리 주소
buffer[8]
..(8바이트)
스택 프레임 포인터
복귀주소
str 주소(함수인자)
function()의 스택
str[512]
..
..(512바이트)
스택 프레임 포인터
복귀주소
함수인자
main()의 스택
높은 메모리 주소

도면 4


310
P
취약점을 가지고 있을
것으로 판단되는
프로그램 목록
320
M
프로그램 목록 P에 속하는
각각의 프로그램이 사용하는
메모리 영역에 대한 정보
330
S
해킹으로 판단되는
취약한 시스템 콜 목록
340
시스템 콜 탐지
엔진 구조
350
공격결정
클래스
370
시스템 콜 주소
추출 모듈
C++
컴파일러
360
시스템 콜 탐지 엔진

도면 5


시작
S401
시스템 콜 호출이 발생하였는가?
아니오
예
시스템 콜 호출 주소 추출
S402
추출된 주소와 정상적인 프로세서 메모리 영역과 비교
S403
S404
시스템 콜 호출이 스택 영역에서 발생하였는가?
아니오
예
해킹으로 판단되는 시스템 콜 목록 S와 비교
S405
시스템 콜 정상 실행
S407
S406
목록 S에 <포함되어 있는 시스템> 콜인가?
아니오
예
프로세서 정지 및 관리자에게 경보
S408
종료